# Machine Learning Methods for Annual Influenza Vaccine Update

by

Lin Tang

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Master of Applied Science
in
Management Sciences

Waterloo, Ontario, Canada, 2013

## AUTHOR'S DECLARATION

# Abstract

Influenza is a public health problem that causes serious illness and deaths all over the world. Vaccination has been shown to be the most effective mean to prevent infection. The primary component of influenza vaccine is the weakened strains. Vaccination triggers the immune system to develop antibodies against those strains whose viral surface glycoprotein hemagglutinin (HA) is similar to that of vaccine strains. However, influenza vaccine must be updated annually since the antigenic structure of HA is constantly mutation.

Hemagglutination inhibition (HI) assay is a laboratory procedure frequently applied to evaluate the antigenic relationships of the influenza viruses. It enables the World Health Organization (WHO) to recommend appropriate updates on strains that will most likely be protective against the circulating influenza strains. However, HI assay is labour intensive and time-consuming since it requires several controls for standardization. We use two machine-learning methods, i.e. Artificial Neural Network (ANN) and Logistic Regression, and a Mixed-Integer Optimization Model to predict antigenic variety. The ANN generalizes the input data to patterns inherent in the data, and then uses these patterns to make predictions. The logistic regression model identifies and selects the amino acid positions, which contribute most significantly to antigenic difference. The output of the logistic regression model will be used to predict the antigenic variants based on the predicted probability. The Mixed-Integer Optimization Model is formulated to find hyperplanes that enable binary classification. The performances of our models are evaluated by cross validation.

**Keywords**: Influenza vaccine update, machine learning, mixed-integer programming, logistic regression, artificial neural network

## Acknowledgements

I would like to express my sincere gratitude to my supervisor Dr. Osman Ozaltin for the continuous support of my Master study and research. He patiently provided the guidance, encouragement and advise necessary for me to proceed through the master program and complete this thesis.

## Table of Contents

## List of Figures

## List of Tables

# 1.0 INTRODUCTION

## 1.1 The Influenza Virus

The influenza viruses are classified into three types: influenza A virus, influenza B virus, and influenza C virus. Type A and B viruses routinely spread in people (human influenza viruses). They cause the annual influenza epidemics that affect up to 20% of the population in the world. The influenza A viruses are subdivided into different serotypes based on the antigenic nature of their surface glycoproteins: haemagglutinin (HA) and neuraminidase (NA). Antigenic drift and antigenic shift are the two processes driving the antigens to change. Antigenic drift is constantly occurring in both types A and B viruses whereas antigenic shift occurs only in type A virus. Antigenic shift refers to the emergence of an entirely new virus subtype that was not circulating among people. Since antigenic shift results in the emergence of a new influenza virus, a large proportion (or even all) of the world's population will have no antibodies against it. Influenza A/H1N1, A/H3N2, and B viruses have been in global circulation since 1977, and these three viruses are currently included in each year's influenza vaccine [18].

## 1.2 Influenza Vaccine Update

To predict the epidemic strains that will prevail in the future flu seasons, a worldwide surveillance network has been set up by the World Health Organization (WHO). Currently, the network consists of 136 national influenza centers in 106 countries and is continually monitoring antigenic drift and other changes (such as antiviral drug resistance) in circulating influenza viruses. Twice a year, the WHO committees meet to consider the data and recommend suitable strains to be included in the influenza vaccine for the subsequent season. This ensures that the vaccine viruses have identical or similar antigenic characteristics to the circulating strains and are effective in preventing the disease [15, 16, 17].

Influenza vaccines are currently produced using embryonated chicken eggs and this process can take up to 9 months. Therefore, the WHO recommends the vaccine strains under uncertainty with partial information. Occurrence of a significant antigenic drift during the lengthy production period can result in a mismatch between the circulating strains and the vaccine strains. This will reduce the effectiveness of the vaccine and result in a potential for an epidemic outbreak.

## 1.3 Data Collection

The vaccine strains are collected by national influenza centers within the framework of the WHO global influenza surveillance network. A crucial mechanism driving the interaction between the virus and the host immune system is cross-immunity: after being infected by a strain, the host acquires partial or total immunity to a set of other strains antigenically similar to the infecting one [22]. The degree of cross–immunity between the two strains can be measured in terms of the Hamming distance between their genetic sequences. In this study, we consider the amino acid sequence of the surface glycoprotein hemagglutinin which has 329 residues. Our dataset consists of a sample of n=31878 pairwise sequence comparison of those 329 residues, taken from Smith et.al [1]. The feature vector is a binary string. For a specific position on the 329 amino acids, "1" indicates a mutation (the residue types of the two sequences on this position are different) while "0" indicates absence of mutation. The dataset also includes an indicator about antigenic variety corresponding to each sample point which will be used as the dependent variables in our model.

## 1.4 Motivation and Contribution

WHO uses a laboratory procedure called hemagglutination inhibition (HI) assay to evaluate the antigenic relationships of circulating influenza viruses. However, HI assay is labour intensive and time-consuming since it requires several controls for standardization. Moreover, the vaccine selection strategy has not been effective in some recent years [22]. This motivates the research for a faster and better strategy for identifying antigenic variety. In this work, we work on the real dataset. We derive two machine-learning methods, i.e. Artificial Neural Network (ANN) and Logistic Regression, and a Mixed-Integer Optimization Model to identify antigenic variance of influenza strains based on the amino acid sequence analysis. The ANN generalizes the input data to patterns inherent in the data, and then uses these patterns to make predictions. The logistic regression model identifies and selects the amino acid positions, which contribute most significantly to antigenic difference. The output of the logistic regression model will be used to predict the antigenic variants based on the predicted probability. The Mixed-Integer Optimization Model is formulated to find hyperplanes that enable binary classification. Our goal is to predict the antigenic drift outcome for new influenza virus strains on the basis of some or all of the amino acid positions for annual vaccine update. And also, we compare the performance of different machine learning and optimization techniques for binary classification.

# 2.0 Literature Review

## 2.1 Hierarchical Clustering

Hierarchical clustering is one of the most straightforward methods to form clusters. It can be either agglomerative or divisive. In the first category, the procedure starts with each object representing an individual cluster. These clusters are then sequentially merged according to some distance measure. In general, to form clusters using a hierarchical cluster analysis, one should select:

- the number of clusters needed
- a criterion for determining distance between objects
- a criterion for determining which clusters are merged at successive steps

On the other hand, divisive methods start with all objects in one cluster. In each step, a cluster is chosen and split up into two. This process continues until n clusters are produced. In this study, our focus will be on the agglomerative clustering.

## 2.2 Artificial Neural Network

ANN is a machine learning technique that can simulate the neurological processing ability of the human brain and can address problems with non-linear and complex data [8]. From a statistical perspective, neural networks have been successfully applied across a wide range of prediction and classification problems [23]. ANNs can be trained to identify correlations in the input data. A multilayer feed-forward neural network is very useful in practice since it can represent a very broad set of nonlinear functions. Also, feed-forward networks are commonly used for classification problems. In this study, we compare the performance of the most popular neural network tool: a feed-forward multilayered network trained using backpropagation, against that of logistic regression and integer programming.

A feed-forward network can be viewed as a graphical representation of a parametric function, which takes a set of input values and maps them to a corresponding set of output values [9]. In feed-forward network, information flows in one direction from the input layer via the hidden layers to the final output layer. Multiple layers of neurons with nonlinear transfer functions allow the network to learn complex relationships between input and output vectors.  We trained our network using standard backpropagation algorithm.

Standard backpropagation is a gradient descent algorithm, which looks for the minimum of the error function in the weight space. The mean sum of squares of the network errors between the network outputs and the target outputs is the typical performance function that is used for training feed forward neural networks. Properly trained backpropagation networks tend to give highly accurate prediction when presented with new input data.

## 2.3 logistic regression

Logistic regression, also called a logit model, is used to model dichotomous outcome variables. In the logit model the log odds of the outcome is modeled as a linear combination of the predictor variables [34]. Since the dependent variable of our problem is dichotomous, a logistic regression model can be built to predict the antigenic variety. The predicted values of our problem are always between 0 and 1, and correspond to the probability of the sequences being 1 (antigenic variant).

## 2.4 Studies Related to Influenza vaccine strain selection

Understanding the antigenic evolution of influenza virus is one of the critical issues in public health. Many methods have been proposed to study the antigenic drift for vaccine development.

To determine the optimal design of influenza vaccine, Ozaltin [21] quantifies the tradeoffs in the optimization of the strain selection decisions arose between the composition of the annual flu shot and the timing of its production. They build two models. The first model takes the view of a social planner, and optimizes strain selections based on a production plan that is provided by the flu shot manufacturers. The second model relaxes the exogenous production planning assumption and, hence, provides a more accurate representation of the hierarchical decision mechanism between a social planner, who designs the flu shot, and the manufacturers, who make the flu shot available.

Lee and Chen [18] investigate the amino acid positions for predicting antigenic variants of influenza A/H3N2 viruses. They build a model based on amino acid differences in the whole HA1 polypeptide (329 residues). It is shown that there is a correlation between the antigenic distance and the number of amino acid changes in the HA1 polypeptide ($R= 0.74$, $p < 0.001$). Different cutoffs of amino acid changes in the HA1 polypeptide are evaluated for predicting antigenic variants. They find that cutoff with more than 7 amino acid changes give the highest agreement (77%).

In practice, not all 329 amino acid positions on the HA1 polypeptide contribute significantly to antigenic difference, some amino acid positions are more important than others. Liao et al. [20] employ stepwise multiple regression and backward conditional logistic regression to select the amino acid positions which are related to antigenic variety. They identify twenty-two amino acid positions with a 92.06% agreement rate.

Huang et al. [19] identify nineteen critical amino acid positions on the HA gene based on the information gain between each amino acid positions. Then they propose rules for predicting antigenic variants using a decision tree.

# 3.0 Classification via Logistic Regression

## 3.1 Model Building

We formulate a logistic regression model to identify critical amino acid positions, which contribute significantly to antigenic variety. After the model refinement and selection, we evaluate the performance of our model using cross validation. The parameters and variables of our model are defined as follows:

*Parameters*

- $p_i$: predicted probability of antigenic drift of sequence $i$ =1,…,31878
- $p^*$ : threshold of logistic regression
- $d_i = 1$ if there is an antigenic drift on sequence $i$ and $d_i = 0$ otherwise for $i$ =1,…,31878

*Variables*

- $y_i = 1$ if $p_i > p^*$and $y_i = 0$ otherwise for $i$ =1,…,31878

In the first step of our approach, we fit a logistic regression model to the training dataset and apply this model to the testing data to get the predicted antigenic drift probabilities $\hat{p}_i$. The logit function of our model is given by:

$$logit(p_i) = \log\left(\frac{p_i}{1-p_i}\right) = \beta_0 + \beta_1 X_{i1} + \cdots + \beta_{329} X_{i329}$$

After calculating the maximum likelihood estimates of the regression coefficients, $\hat{\beta}$, the predicted probabilities for sequences are given by:

$$\hat{p}_i = \left[1 + \exp\left(-\sum_{j=0}^{329} \hat{\beta}_j X_{ij}\right)\right]^{-1}$$

where $X_{i0} = 1$.

In the second step, predictions can be made for the testing set by using a threshold $p*$ on the output probability of the model.

### 3.1.1 Variable Selection and Performance Measurement

We randomly split our data into 10 training sets and 10 testing sets and build a logistic regression model for each training set. For logistic regression models, it is possible to test the statistical significance of the coefficients in the model by using a z-statistic (sometimes called a Wald z-statistic), and the associated p-values. We start with the full model and at each step remove those variables which fail to meet the threshold of p < 0.1, and build the second model with only the significant variables. The process iterates until all variables are significant. For each training set, we identify all critical amino acid positions and build our final model with only significant variables. 10-fold cross validation is used to validate how well our model learns from the training data. We use the classification matrix to evaluate the accuracy of our model's prediction. If the predicted and actual value is the same, i.e. 1 and 1 or 0 and 0, then the prediction is accurate for that case. If predicted value and actual value are different, the model is not accurate for that case. Agreement rate, sensitivity and specificity are calculated for each fold. Sensitivity measures the proportion of actual positives which are correctly identified as positive. Specificity measures the proportion of negatives which are correctly identified as negative. A perfect predictor would be described as 100% sensitivity and 100% specificity. We use Receiver operating characteristic (ROC) curve to illustrate the performance of our model. Table 1 summarizes the results of the analysis for both training and testing dataset.

**Table 1 Classification accuracy of logistic regression**

| **Model** | **No. of Selected positions** | **Testing dataset** | | | **Training dataset** | | |
|---|---|---|---|---|---|---|---|
| | | Sensitivity | Specificity | Agreement Rate | Sensitivity | Specificity | Agreement Rate |
| 1 | 18 | 0.992 | 0.964 | 0.988 | 0.992 | 0.956 | 0.987 |
| 2 | 15 | 0.990 | 0.952 | 0.985 | 0.991 | 0.955 | 0.985 |
| 3 | 15 | 0.991 | 0.962 | 0.987 | 0.989 | 0.954 | 0.984 |
| 4 | 18 | 0.993 | 0.964 | 0.988 | 0.992 | 0.955 | 0.986 |
| 5 | 17 | 0.991 | 0.942 | 0.984 | 0.992 | 0.957 | 0.986 |
| 6 | 19 | 0.990 | 0.962 | 0.986 | 0.991 | 0.961 | 0.987 |
| 7 | 18 | 0.994 | 0.962 | 0.989 | 0.991 | 0.955 | 0.986 |
| 8 | 17 | 0.991 | 0.958 | 0.986 | 0.991 | 0.964 | 0.987 |
| 9 | 14 | 0.991 | 0.970 | 0.988 | 0.991 | 0.954 | 0.986 |
| 10 | 21 | 0.990 | 0.957 | 0.985 | 0.991 | 0.960 | 0.986 |
| Average | 17 | 0.991 | 0.959 | 0.987 | 0.991 | 0.957 | 0.986 |

We observe that model 7 has the highest agreement rate for both training and testing datasets. Its high sensitivity rate helps identify antigenic variants more easily. The high specificity rate indicates a great accuracy of identifying similar viruses. In model 7, we identify 18 positions as critical amino acid positions, which are positions: 63, 133, 137, 143, 144, 145, 156, 158, 172, 189, 190, 193, 197, 208, 214, 219, 275 and 299. We build our final model, with all significant variables (all critical amino acid positions). Figure 1 and Figure 2 show the ROC for training and testing sets of model 7, respectively.

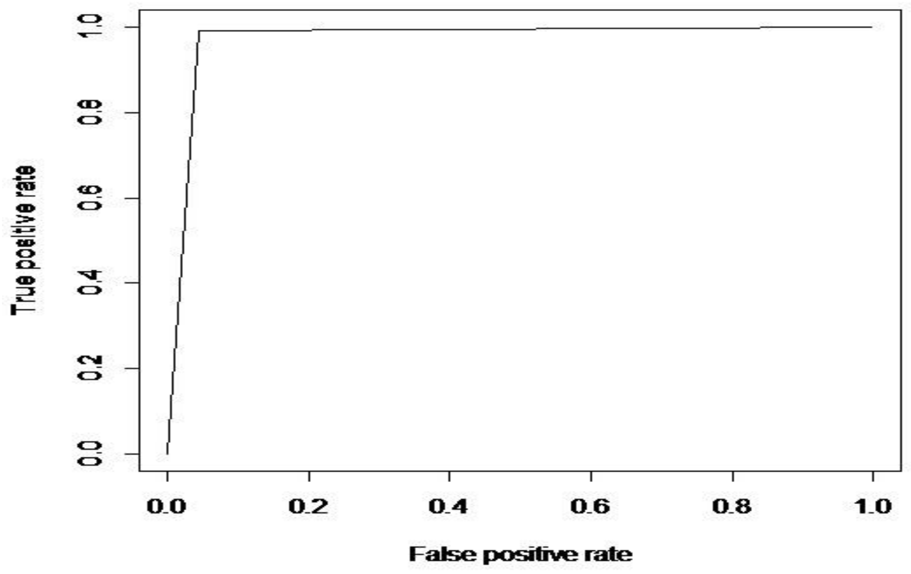

**Figure 2 ROC plot for testing dataset**

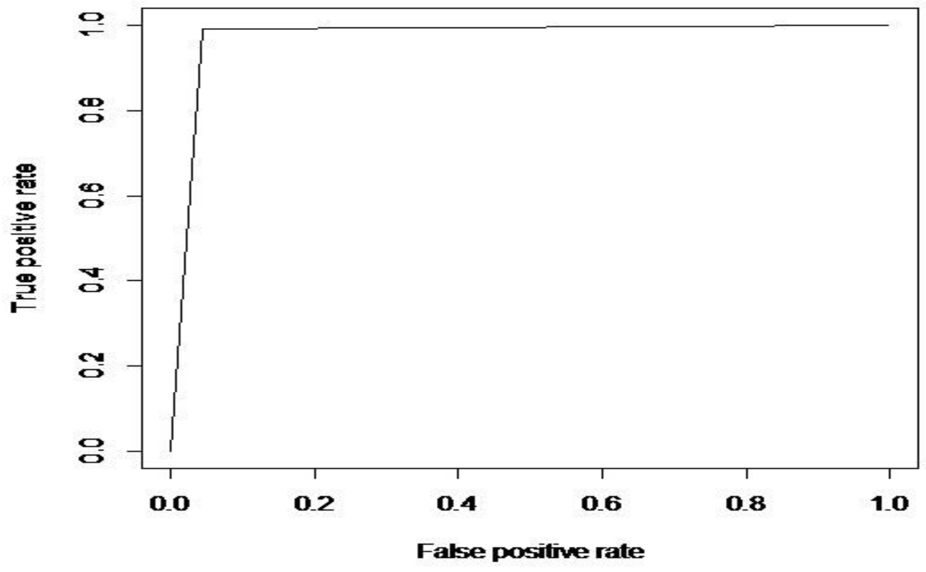

**Figure 3 ROC plot for Training dataset**

From the summary output of the final model, as the p-values of 18 variables are less than 0.1, they are all statistically significant in the logistic regression model. The null deviance for the logistic model fit to the final model data is 24250.2 on 28690 degrees of freedom. The residual deviance is 2214.2 on 28672 degrees of freedom. The deviance is reduced by 22036 points on 18 degrees of freedom, with a p-value of almost 1. Large P-value means we have no/very weak evidence against our null hypothesis. In this case the null hypothesis states that the logistic regression model provides an adequate fit to the data. The residual deviance of 2411 is small with a p-value of almost 0, which indicates a good fit of the logistic model.

ROC shows the tradeoff between sensitivity and specificity (any increase in sensitivity will be accompanied by a decrease in specificity). The area under curve (AUC) provides a measure of the model's ability to discriminate between those with and without antigenic variants. From Figure 1 and Figure 2, we observe that the curve follows very close to the left-hand border and the top border of the ROC space. This means the AUC is very close to 1 and the prediction accuracy is very high.

# 4.0 Classification via Integer Programming

## 4.1 Classification Problems

We apply a two-class classification problem proposed by Bertsimas and Shioda [3] with input sequence $x_i$, i =1,…,n, where n is the size of the training set. Each sequence is a pattern of 329 amino acid positions. The indicator variables $y_i \in \{0,1\}$, $i$ =1,⋯,$n$, where $y_i = 1$ if there is an antigenic variant and 0 otherwise. Therefore, we identify the two classes with the binary value 1 and 0. We classify the sequences with antigenic drift ($y_i = 1$) as Class 1 sequences and sequences with the same genetic characteristics ($y_i = 0$) as Class 0 sequences. Let $n_0$ and $n_1$ be the number of Class 0 and Class 1 sequences, respectively. Let $N_0 = \{1,…,n_0\}$ and $N_1 = \{1,…,n_1\}$. Class 0 and Class 1 sequences are denoted by $x_i^0, i = 1, \dots, n_0$, and $x_i^1, i = 1, \dots, n_1$, respectively. Our goal is to partition Class 1 sequences into K disjoint groups such that no class 0 strain can be expressed as a combination of Class 1 sequences. Then new sequences can be classified as Class 1 if it belongs to one of these groups or as Class 0 otherwise. Let $\bar{K} = \{1, \dots, K\}$ and $G_k$ be the set indices of Class 1 sequences that are in group k, where $\bigcup_{k\in\bar{K}} G_k = N_1$ and $G_k \cap G_{k'} = \emptyset, k, k' \in \bar{K}, k \neq k'$. Variables and parameters are defined in Table 2.

**Table2 Definition of Variables and Parameters**

| **Variables** | **Definition** |
|---|---|
| $x_i$, $i$=1,…,n | Pattern of 329 amino acid positions |
| $y_i$, $i$=1,…,n | Indicator variables $y_i = 1$ if there is a antigenic variant and 0 otherwise |
| $n_0$ | Number of Class 0 strain |
| $n_1$ | Number of Class 1 strain |
| $x_i^0, i = 1, \dots, n_0$ | Class 0 strains |
| $x_i^1, i = 1, \dots, n_1$ | Class 1 strains |
| $N_0$ | Set of indices of Class 0 strains $N_0 = \{1, \dots, n_0\}$ |
| $N_1$ | Set of indices of Class 1 strains $N_1 = \{1, \dots, n_1\}$ |
| $K$ | Number of disjoint groups to which Class 1 sequences will be classified |
| $\bar{K}$ | Set of indices of groups $\bar{K} = \{1, \dots, k\}$ |
| $G_k$ | Set indices of Class 1 strains that are in group k |

To ensure that there is no Class 0 sequence in any of the Class 1 group, model (1) has to be infeasible for all $i \in N_0$ *and* $k \in \overline{K}$:

$$\sum_{j \in G_k} \lambda_j x_j^1 = x_i^0,$$

$$\sum_{j \in G_k} \lambda_j = 1, \qquad (1)$$

$$\lambda_j \geq 0, j \in G_k.$$

From Farkas' Lemma [4], model (1) has no solution if and only if:

$$p'x_i^0 + q < 0,$$

$$p'x_j^1 + q \geq 0, j \in G_k \qquad (2)$$

We check the feasibility of model (2) using the optimization problem:

$$z_{k,i} = max\ \varepsilon$$

$$\textbf{s.t}\ p'x_i^0 + q \leq -\varepsilon, \qquad (3.1)$$

$$p'x_j^1 + q \geq 0,\ j \in G_k \qquad (3.2) \qquad (3)$$

$$\varepsilon \leq 1. \qquad (3.3)$$

Model (3) determines if there exists a hyperplane $\boldsymbol{p'x} + q = 0$ that separates sequence $\boldsymbol{x_i^0}$ from all Class 1 sequences in group *k*. If $z_{k,i}$>0, then model (2) is feasible, which also implies that model (1) is infeasible. If $z_{k,i}$ ≤0, model (2) is infeasible, as a result model (1) is feasible. The constraint $\varepsilon \leq 1$ is added to prevent unbounded solutions.

We expand model (3) for all $k \in \overline{K}$ and $i \in N_0$.

$$z = max\ \delta$$

$$\textbf{s.t}\ p'_{k,i}x_i^0 + q_{k,i} \leq -\delta, i \in N_0; k \in \overline{K} \qquad (4.1)$$

$$p'_{k,i}x_j^1 + q_{k,i} \geq 0, i \in N_0; k \in \overline{K}, j \in G_k \qquad (4.2) \qquad (4)$$

$$\delta \leq 1. \qquad (4.3)$$

We define decision variable $a_{k,j}$ , $k \in \overline{K}$ , $j \in N_1$ to decide if we can assign Class 1 sequence *j* to group *k*,

$$a_{k,j} = \begin{cases} 1 & if\ x_j^1 is\ assigned\ to\ group\ k \\ 0 & otherwise \end{cases}$$

To ensure that we enforce constraint (3.2) of model (3) if $a_{k,j} = 1$, we modify the constraint (4.2) as: $p'_{k,i}x_j^1 + q_{k,i} \geq a_{k,j} - 1$.

The model becomes:

$$
\begin{aligned}
& z^* = max\,\delta \\
& \textbf{s.t } p'_{k,i}x_i^0 + q_{k,i} \leq -\delta, i \in N_0; k \in \bar{K} && (5.1) \\
& p'_{k,i}x_j^1 + q_{k,i} \geq a_{k,j} - 1, i \in N_0; k \in \bar{K}, j \in G_k && (5.2) \qquad (5) \\
& \textstyle\sum_{k=1}^{K} a_{k,j} = 1, j \in N_1 && (5.3) \\
& \delta \leq 1 && (5.4) \\
& a_{k,j} \in \{0,1\}. && (5.5)
\end{aligned}
$$

## 4.2 The Clustering Algorithm

By solving model (5), we can check whether Class 1 sequences can be partitioned into $K$ groups without including any of the Class 0 sequences in their convex hull. However, finding a hyperplane for each of the Class 0 sequences is computationally expensive. As an alternative, we find a hyperplane for a cluster of Class 0 sequences at a time. We need an algorithm to cluster our Class 1 and Class 0 sequences to $K_1$ and $K_0$ clusters, respectively.

We apply the hierarchical (agglomerative) clustering procedures to create clusters of Class 1 and Class 0 sequences. Initially, we start with each Class 1 (Class 0) sequence representing an individual cluster. Then two clusters with the smallest statistical distance between them are merged. We stop merging when none of the clusters can be further merged. During the procedure, we need to ensure that a merger of Class 1 (Class 0) clusters will not contain any Class 0 (Class 1) sequences in the resulting convex hull. We include the following optimization problem in our clustering algorithm to check whether Class 1 clusters *r* and *s* can be merged:

$$
\begin{aligned}
& \delta^* = \max \delta \\
& s.t\ p'_i x_i^0 + q_i \leq -\delta,\ i \in N_0, && (6.1) \\
& p'_i x_j^1 + q_i \geq \delta,\ j \in C_r \cup C_s, && (6.1) \qquad (6)
\end{aligned}
$$

where $C_r$ and $C_s$ are the sets of indices of Class 1 sequences in clusters r and s, respectively.

If $\delta^* > 0$, problem (6) is feasible, which means clusters r and s can be merged. If $\delta^* \le 0$, problem (6) is infeasible, indicating that clusters r and s cannot be merged. We present an outline of the algorithm as follows:

**Algorithm 1:**

```
1.  Initialize: $K = n_1, k = 0$.
2.  While $k < K$ do
3.  Find the clusters with minimum pairwise distance, call these r and s.
4.  Solve problem (6) on clusters r and s.
5.  If $\delta^* \le 0$, then
6.      k=k+1
7.  else
8.      Merge clusters r and s.
9.      K=K-1, k=0
10. end if
11. end while
```

Initially, each Class 1 sequence represents an individual cluster, $K = n_1$. We can take any random sequence as the initial centroids. We obtain the minimum pairwise distance on line 3 by comparing the statistical distances between the centroids of all clusters.

**Definition 4.1**: The statistical distances between two points $x = (x_1, \dots, x_d)^t$ and $y = (y_1, \dots, y_d)^t$ in the d-dimensional space $\mathbb{R}^d$ is defined as $d_{\mathbb{R}}(x, y) = \sqrt{\sum_{j=1}^{d} \frac{(x_j - y_j)^2}{s_j^2}}$, where $s_j^2$ is the sample variance of $j^{th}$ coordinate of all points in $\mathbb{R}$.

For testing purpose, we divide our data into 10 training and testing sets. For training set 1, we get $K_1 = 114$ clusters of Class 1 sequences using **Algorithm 1** and $K_0 = 187$ clusters of Class 0 sequences following the same procedure.

An outlier is an observation of the data that deviates from other observations so much that it arouses suspicions that it was generated by a different mechanism from the most part of data [5]. In clustering, outliers are considered as observations that should be removed in order to make clustering more reliable [6]. In outlier detection methods based on clustering, outlier is defined to be an observation that does not fit to the overall clustering pattern [7]. After $K_0$ and $K_1$ clusters are obtained, we eliminated all clusters with cardinality less than 1% of $n_0$ and $n_1$, respectively. After the outlier removal procedure, we get 25 clusters of Class 0 sequences and 7 clusters of Class 1 sequences. Since there are only 7 clusters of Class 1 sequences, we will consider each cluster as a group, which means $K = K_1$.

## 4.3 Assigning Groups to Polyhedral Regions

To separate Class 0 and Class 1 sequences, Bertsimas and Shioda [3] present an approach using hyperplanes such that the minimum Euclidean distance from any point to the hyperplane is maximized. For each group $k, k \in K$, of Class 1 sequences and for every cluster $t, t \in K_0$, of Class 0 sequences, we find a hyperplane $\pi'_{k,t} = \alpha_{k,t}$. Our objective is to maximize the minimum distance between each sequence and the hyperplane. The distance is defined as:

$$d(x, \pi_{k,t}, \alpha_{k,t}) = \frac{|\gamma|}{||\pi_{k,t}||}, where\, \gamma = \pi'_{k,t}x - \alpha_{k,t}.$$

By fixing $|\gamma|$ and minimizing$||\pi_(k,t)\ ||^2$, the following quadratic optimization problem maximizes the distance:

Min $\pi'_{k,t}\pi_{k,t}$

s.t $\pi'_{k,t}x_i^0 \geq \alpha_{k,t} + 1.\, i \in C_t^0,$ (8.1) (8)

$\pi'_{k,t}x_j^1 \geq \alpha_{k,t} - 1.\, i \in G_k.$ (8.2)

By solving problem (8) for each group K, we obtained $KK_0$ hyperplanes. The polyhedral region for each group *k* is

$$P_k = \left\{x \in \mathbb{R}^d \middle| \pi'_{k,t}x \leq \alpha_{k,t}, t \in K_0\right\}.$$

However, some of the hyperplanes may be redundant. To check the redundancy of the hyperplanes, we solve the following problem:

$$\omega_{k,t_0} = max\ \pi'_{k,t_0}x$$

$$\text{s.t}\quad \pi'_{k,t}x \le \alpha_{k,t}, \qquad i \in C_t^0, \qquad (9.1) \qquad (9)$$

$$\pi'_{k,t_0}x \le \alpha_{k,t_0} + 1.\, i \in G_k. \qquad (9.2)$$

If $\omega_{k,t_0} \le \alpha_{k,t_0}$, the corresponding hyperplane is redundant. Since this problem needs to be solved for each group k, $k \in K$ and for every cluster $t,\ t\epsilon K_0$, we use the following algorithm to eliminate all redundant hyperplanes:

```
1. for $k$=1 to $K$ do
2.     for $t_0$=1 to $K_0$ do
3.         Solve problem (9).
4.         if $\omega_{k,t_0} \le \alpha_{k,t_0}$ then
5.             Eliminate constraint $\pi'_{k,t_0}x \le \alpha_{k,t_0}$.
6.         end if
7.     end for
8. end for
```

We find that none of the hyperplanes are redundant. After we have the polyhedra, we can test new data with this model. If the sequence lies in any of the *K* polyhedra, we classify the sequence as Class 1 sequence. If the sequence is not contained in any of the polyhedra, then we classify this sequence as Class 0 sequence.

## 4.4 Performance Measurement

We run the same procedure for all training sets. The training and testing results of our optimization model are shown in Table 3.

**Table 3 Classification accuracy of integer programming**

| **Dataset** | **Testing dataset** | | | **Training dataset** | | |
|---|---|---|---|---|---|---|
| | Sensitivity | Specificity | Agreement Rate | Sensitivity | Specificity | Agreement Rate |
| 1 | 0.999 | 0.992 | 0.997 | 0.998 | 0.992 | 0.997 |
| 2 | 0.893 | 0.952 | 0.902 | 0.885 | 0.938 | 0.893 |
| 3 | 0.856 | 0.916 | 0.865 | 0.839 | 0.937 | 0.853 |
| 4 | 0.981 | 0.462 | 0.903 | 0.978 | 0.494 | 0.906 |
| 5 | 0.949 | 0.653 | 0.904 | 0.949 | 0.666 | 0.907 |
| 6 | 0.960 | 0.753 | 0.929 | 0.956 | 0.747 | 0.925 |
| 7 | 0.980 | 0.412 | 0.895 | 0.980 | 0.415 | 0.895 |
| 8 | 0.887 | 0.663 | 0.853 | 0.880 | 0.678 | 0.849 |
| 9 | 0.915 | 0.688 | 0.881 | 0.922 | 0.691 | 0.887 |
| 10 | 0.976 | 0.661 | 0.929 | 0.980 | 0.657 | 0.932 |
| Average | 0.939 | 0.715 | 0.906 | 0.937 | 0.721 | 0.904 |

From Table 3, we observe that the sensitivity rates for both datasets are very high. This indicates that our model can accurately identify antigenic variants. However, the specificity rates are relatively low. A test with a high sensitivity but low specificity results in many sequences which have no antigenic variants being predicted as antigenically different.

# 5.0 Classification via Artificial Neural Network

ANNs have been recently applied to clinical problems such as diagnosing myocardial infarcts [11] and breast cancer detection [12]. We develop a method of classifying antigenic variant using Artificial Neural Networks. We train a neural network on amino acid sequences to test the NN's accuracy of prediction on new sequences. The network receives the 329 binary values as an input vector. It is then required to identify the antigenic variant by responding an output value between [0, 1].

## 5.1 Network Architecture

Deciding the number of hidden layer and number of neurons in each of these hidden layers are critical steps for deciding the overall neural network architecture. Even though the layers do not directly interact with the external environment, these layers have a tremendous influence on the final output.

We first determine the number of hidden layers to use with the neural network. The most common network used with backpropagation is the two-layer feedforward network [10]. Two-layer feed-forward networks can represent any input-output relationship with a finite number of discontinuities. It gives a general result showing that nonlinear control systems can be stabilized using two hidden layers, but not in general using just one [13]. For most problems, it starts with two layers, and then increase to three layers if the performance with two layers is not satisfactory. However, there is currently no mathematical theory that provides a definitive answer to the choice of number of hidden neurons in each of these hidden layers. If too few hidden neurons are used, the network will be unable to model complex data, resulting in underfitting. Too many hidden neurons can increase the training time dramatically and result in overfitting. We start with a two-layer model with 20 neurons in each hidden layer.

### 5.1.1 Activation Functions

The sigmoid activation function is a common activation function for neural networks. It constrains the outputs of a network to be between 0 and 1. The sigmoid activation function is most useful for training data that is also between 0 and 1. Since our input and target (output) data are all binary, we consider a sigmoid transfer function for the hidden and output layers.

## 5.2 Comparison of Training Algorithm

*Trainlm* is the fastest training function and the default training function for feedforward network. The quasi-Newton method, *trainbfg* and *trainoss* are also quite fast since they do not need to store the large

Hessian matrix. All of these methods tend to be less efficient for large networks (with thousands of weights), since they require more memories and computation time for these cases. Also, *trainlm* performs better on function fitting (nonlinear regression) problems than on pattern recognition problems [10]. When training large networks or pattern recognition networks, the conjugate gradient method, *traincgp, trainscg* and *traincgf* are good choices. Their memory requirements are relatively small, and yet they are much faster than standard gradient descent algorithms (*traingd*, *traingda, traingdx*). *Trainrp* is a network training function that updates weight and bias values according to the resilient backpropagation algorithm.

### 5.2.1 Train and Test Strategy

In order to train networks to find the necessary input output relationships without over-fitting the training dataset, we apply the early-stopping technique [14]. The dataset is divided into three sets: training, validation and testing set. In early stopping, the training set is split into a new training set and a validation set. We train the networks with the training data and evaluate the validation error at each iteration. When the performance of the validation test stops improving or a specified number of iterations (which is set to 1000 in our study) is reached, the algorithm halts. The network with the best performance on the validation set is then used for actual testing on a separate set of data. We apply 10-fold cross validation to validate how well our model learns from the training data.

### 5.2.2 Result and Discussion

We compare the performance of various classification algorithms in terms of their predictive abilities and computation efficiencies. We use the sensitivity and specificity rates on both training and testing set as the measure of predictive accuracy, and the computation time (in CPU seconds) as the measure of computation efficiency. Table 4 summarizes the average computation time of different algorithms. Table 5 summarizes the ANNs' training performance in terms of accuracy of prediction.

**Table 4 Comparison of CPU Time using different training algorithms**

| Function | Algorithm | Mean(s) | Min(s) | Max(s) | Std.(s) |
|---|---|---|---|---|---|
| trainbfg | Quasi-Newton | 11527 | 5915 | 24434 | 5261 |
| traincgf | Fletcher-Powell Conjugate Gradient | 765.57 | 462.1 | 1163.5 | 213.23 |
| traincgp | Polak-Ribiére Conjugate Gradient | 308.87 | 176.25 | 468.46 | 97.597 |
| traingd | Gradient Descent | 1643.9 | 1634.4 | 1650.6 | 5.5985 |
| traingda | Adaptive Learning Rate Gradient Descent | 109.33 | 70.481 | 188.68 | 32.349 |
| traingdx | Variable Learning Rate Gradient Descent | 122.41 | 83.429 | 169.42 | 24.978 |
| trainlm | Levenberg-Marquardt | 3163.7 | 1007.3 | 8692.3 | 2548.1 |
| trainoss | One Step Secant | 465.78 | 256.7 | 1065.7 | 231.01 |
| trainrp | Resilient Backpropagation | 311.77 | 191.58 | 438.35 | 90.346 |
| trainscg | Scaled Conjugate Gradient | 423.65 | 207.29 | 663.58 | 151.06 |

**Table 5 Comparison of results using different training algorithms**

| **Function** | **Algorithm** | **Testing dataset** | | | **Training dataset** | | |
|---|---|---|---|---|---|---|---|
| | | Sensitivity | Specificity | Agreement Rate | Sensitivity | Specificity | Agreement Rate |
| trainbfg | Quasi-Newton | 0.981 | 0.997 | 0.995 | 0.982 | 0.997 | 0.995 |
| traincgf | Fletcher-Powell Conjugate Gradient | 0.982 | 0.997 | 0.995 | 0.988 | 0.998 | 0.996 |
| traincgp | Polak-Ribiére Conjugate Gradient | 0.997 | 0.980 | 0.995 | 0.997 | 0.983 | 0.995 |
| traingd | Gradient Descent | 0.987 | 0.919 | 0.977 | 0.987 | 0.918 | 0.976 |
| traingda | Adaptive Learning Rate Gradient Descent | 0.971 | 0.862 | 0.955 | 0.972 | 0.860 | 0.955 |
| traingdx | Variable Learning Rate Gradient Descent | 0.995 | 0.958 | 0.989 | 0.996 | 0.957 | 0.990 |
| trainlm | Levenberg-Marquardt | 0.997 | 0.983 | 0.995 | 0.998 | 0.989 | 0.996 |
| trainoss | One Step Secant | 0.997 | 0.980 | 0.995 | 0.997 | 0.982 | 0.995 |
| trainrp | Resilient Backpropagation | 0.957 | 0.948 | 0.955 | 0.957 | 0.953 | 0.956 |
| Trainscg | Scaled Conjugate Gradient | 0.997 | 0.985 | 0.995 | 0.998 | 0.987 | 0.996 |

Comparing all the different training algorithms, the agreement rates are very close while the training time significantly differs. From table 4, we observe that *traingda*, *traingdx* and *traincgp* outperform the other algorithms in terms of mean and standard deviation of training time. *Traingda* and *traingdx* have relatively lower agreement rate than the other algorithms. *Traincgp* is the third fastest algorithm after the *traingda* and *traingdx*, with the agreement rates of 0.995 for both training and validation datasets. The extremely high sensitivity rates of 0.997 for both datasets help us to identify antigenic variants more easily. The specificity rates of the training and validation datasets are 0.983 and 0.980, respectively. This indicates a high accuracy of identifying similar viruses. The best result in terms of both classification agreement rate and training time is obtained using *traincgp*. The performance plot and trainstate plot of *traincgp* are shown in Figure 3 and Figure 4, respectively.

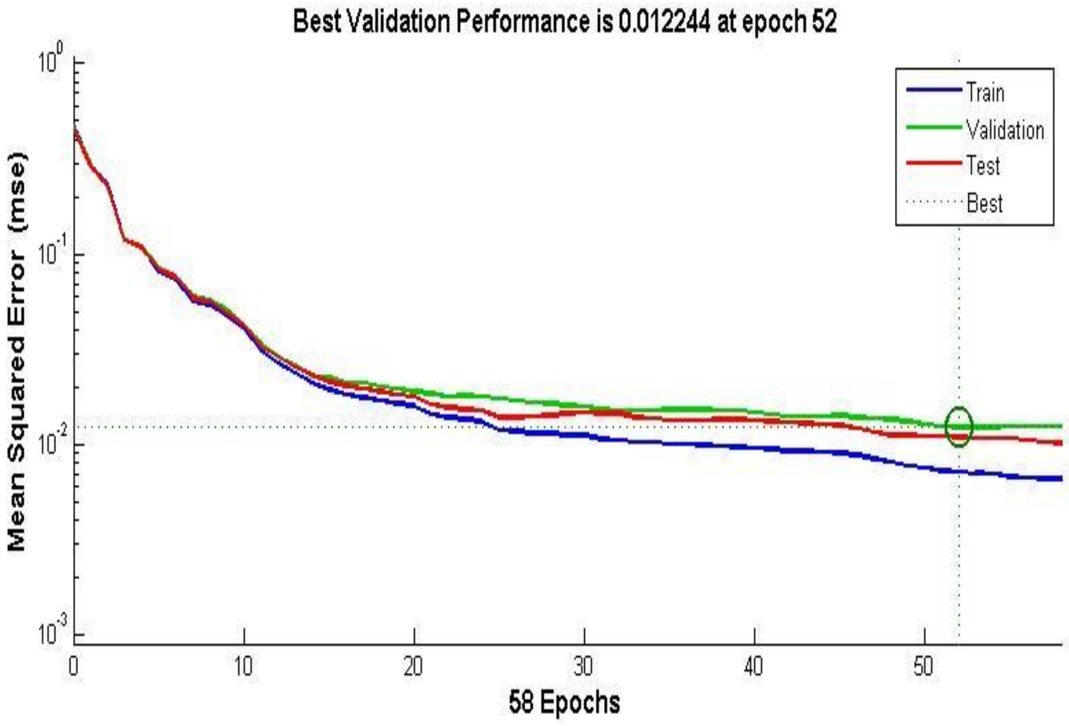

**Figure 3 Performance Plot using *traincgp***

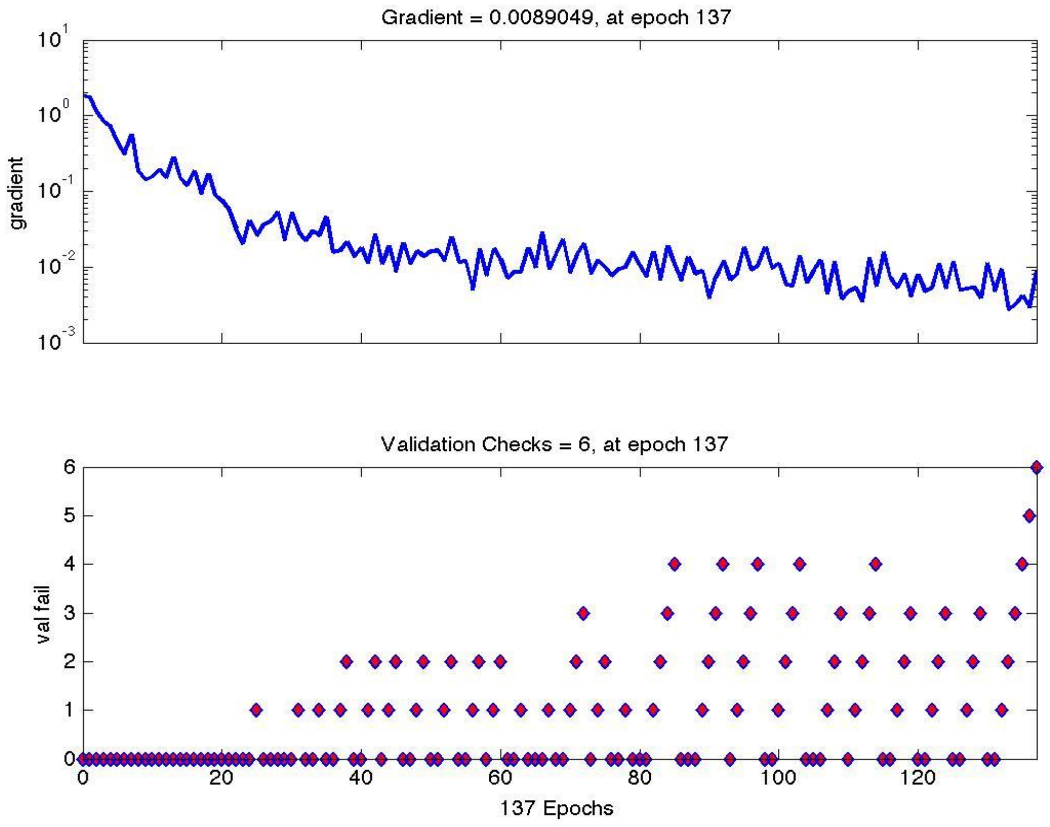

**Figure 4 Trainstate Plot using *traincgp***

The performance of classification (Figure 3) shows the gradual reduction of mean square error (mse) values epoches after epoches. The result is reasonable since the final MSE is small, the test and validation set error have similar characteristics, and it doesn’t appear that any significant overfitting has occurred until epoch 52. From this observation, we can conclude that our model learns from the training data to map the input and output parameters at epoch 52. The training state plot (Figure 4) shows that the gradient will become very small as the training reaches the minimum of the performance. If the magnitude of the gradient is less, the training will stop. The gradient decreased to 0.0089049 at epoch 137. The number of validation checks represents the number of successive iterations that the validation performance fails to decrease. If this number reaches 6 (the default value), the training will stop. The validation checks increase rapidly to 6 at epoch 137.

# 6.0 Summary and Future Research

In this study, we aim on speeding up influenza strain selection process using integer programming and machine learning methods.

In chapter 3, we build a logistic regression model in R 2.15.1 to identify and select the antigenic critical amino acid positions, which contribute most significantly to antigenic variants. After the model refinement and selection, of the 329 amino acids of HA, we identify 18 positions as critical amino acid positions. Based on the final model built with all critical amino acid positions, the accuracies of our model are 98.9% and 98.6% for training and testing set, respectively. The identified critical amino acid positions are similar to relevant works. We believe our method is efficient for vaccine strain update.

In chapter 4, we formulate an integer optimization model and solve the integer programming in IBM ILOG CPLEX® 12.4 to define a set of polyhedra. These polyhedra can be used to predict the class of new sequences. If a sequence lies in any of the *K* polyhedra, we classify this sequence as Class 1, otherwise, Class 0. The agreement rates are high for both training and validation sets. But the model we formulate, in common with many reported by previous studies (Lee and Chen,2004; Liao et al., 2008; Huang et al.,2009) can correctly identify the vast majority of sequences with antigenic variant (high sensitivity) but tend to overpredict the number of sequences with antigenic variant (relatively lower specificity) [1]. Due to its use of integer programming, another shortcoming of classifying through integer optimization model is the computation time. Compare to the machine learning method such as logistic regression and neural network, this takes much longer running time.

In chapter 5, we introduce a method of classifying antigenic variants using Artificial Neural Networks (via MATLAB®'s neural network toolbox). We train a neural network on amino acid sequences using *traincgp* algorithm and test the NN's accuracy of prediction on new sequences. Based on its classification accuracy, neural network outperforms both logistic regression and integer programming methods. However, the major weakness of neural network is its lack of interpretability. The results of an ANN cannot be explained since it is a "black-box" with a set of weights with no inherent meaning.

# Bibliography

[1] Smith, D. J., Lapedes, A. S., de Jong, J. C., Bestebroer, T. M., Rimmelzwaan, G. F., Osterhaus, A. D. & Fouchier, R. A. (2004). Mapping the antigenic and genetic evolution of influenza virus. Science, 305, 371–376.

[3] Bertsimas, D., Shioda, R., 2007. Classification and regression via integer optimization. Operations Research 55 (2), 252-271.

[4] Vanderbei, R. J. (2001) Linear Programming, Foundations and Extensions (Kluwer Academic, Boston).

[5] G. Williams, R. Baxter, H. He, S. Hawkins and L. Gu, “A Comparative Study for RNN for Outlier Detection in Data Mining”. In Proceedings of the 2nd IEEE International Conference on Data Mining, page 709, Maebashi City, Japan, December 2002.

[6] Guha, S., Rastogi, R., Shim, K.: CURE an efficient clustering algorithm for large databases. In: Proceedings of the 1998 ACM SIGMOD International Conference on Management of Data, Seattle, Washington (1998) 73–84

[7] Zhang, T., Ramakrishnan, R., Livny, M.: BIRCH: A new data clustering algorithm and its applications. Data Mining and Knowledge Discovery 1 (1997) 141–182

[8] Achanta AS, Kowalski JG, Rhodes CT. Artificial neural net-works: implications for pharmaceutical sciences. Drug Dev Ind Pharm. 1995;21(1):119-155.

[9] Bishop, C. M. (1995). Neural Networks for Pattern Recognition. Oxford University Press.

[10] BealeM, DemuthH.NeuralNetworks Toolbox forUse with Matlab: User’s Guide Version 3. Natick, Mass: The MathWorks; 1998.

[11] Heden B, Ohlin H, Rittner R, Edenbrandt L. Acute myocardial infarction detected in the 12-lead ECG by artificial neural networks. Circulation. 1997;96:1798–1802

[12] Shekhar Singh., and P. R. Gupta., “Breast Cancer detection and Classification using Neural Network”, International Journal of Advanced Engineering Sciences and Technologies, Vol. 6, No. 1, pp.4 – 9, 2011.

[13] E. Sontag. Feedback stabilization using two-hidden-layer nets. IEEE Trans. Neural Networks,3:981-990, 1992.

[14] M.T. Hagan, H.B. Demuth, and M.H. Beale, Neural Network Design, Boston: PWS Pub. Co., 1995.

[15] Glezen WP. Emerging infections: pandemic influenza. Epidemiol Rev 1996; 18(1):64–76.

[16] P.A. Gross, A.W. Hermogenes, H.S. Sacks, J. Lau, R.A. Levandowski. The efficacy of influenza vaccine in elderly persons A meta-analysis and review of the literature. Ann Intern Med, 123 (7) (1995), pp. 518–527

[17] J. Nordin, J. Mullooly, S. Poblete, R. Strikas, R. Petrucci, F. Wei et al. Influenza vaccine effectiveness in preventing hospitalizations and deaths in persons 65 years or older in Minnesota, New York, and Oregon: data from 3 health plans. J Infect Dis, 184 (6) (2001), pp. 665–670

[18] M.S. Lee, J.S. Chen. Predicting antigenic variants of influenza A/H3N2 viruses. Emerg Infect Dis, 10 (8) (2004), pp. 1385–1390

[19] Huang, J. W., Chen, C. C., & Yang, J. M. (2008, May). Identifying critical positions and rules of antigenic drift for influenza A/H3N2 viruses. In Bioinformatics and Biomedical Engineering, 2008. ICBBE 2008. The 2nd International Conference on (pp. 249-252). IEEE.

[20] Liao, Y. C., Lee, M. S., Ko, C. Y., Hsiung, C. A. (2008). Bioinformatics models for

predicting variants of influenza A/H3N2 viruses. *Bioinformatics* 24:505–512.

[21] Ozaltin, O. Y. (2012). *Optimal design of the annual influenza vaccine* (Doctoral dissertation, UNIVERSITY OF PITTSBURGH).

[22] Wu, J. T., Wein, L. M., & Perelson, A. S. (2005). Optimization of Influenza Vaccine Selection. *Operations Research*, *53*(3).

[23] Zhang, G. P. (2000). Neural networks for classification: a survey. Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on, 30(4), 451-462.